DragNDrop Minimap

Overview

Video Tutorial: https://www.youtube.com/playlist?list=PLzm69oEaPR_eSTfcvyIni3TpoNpBtBiaE

Demo Scene:

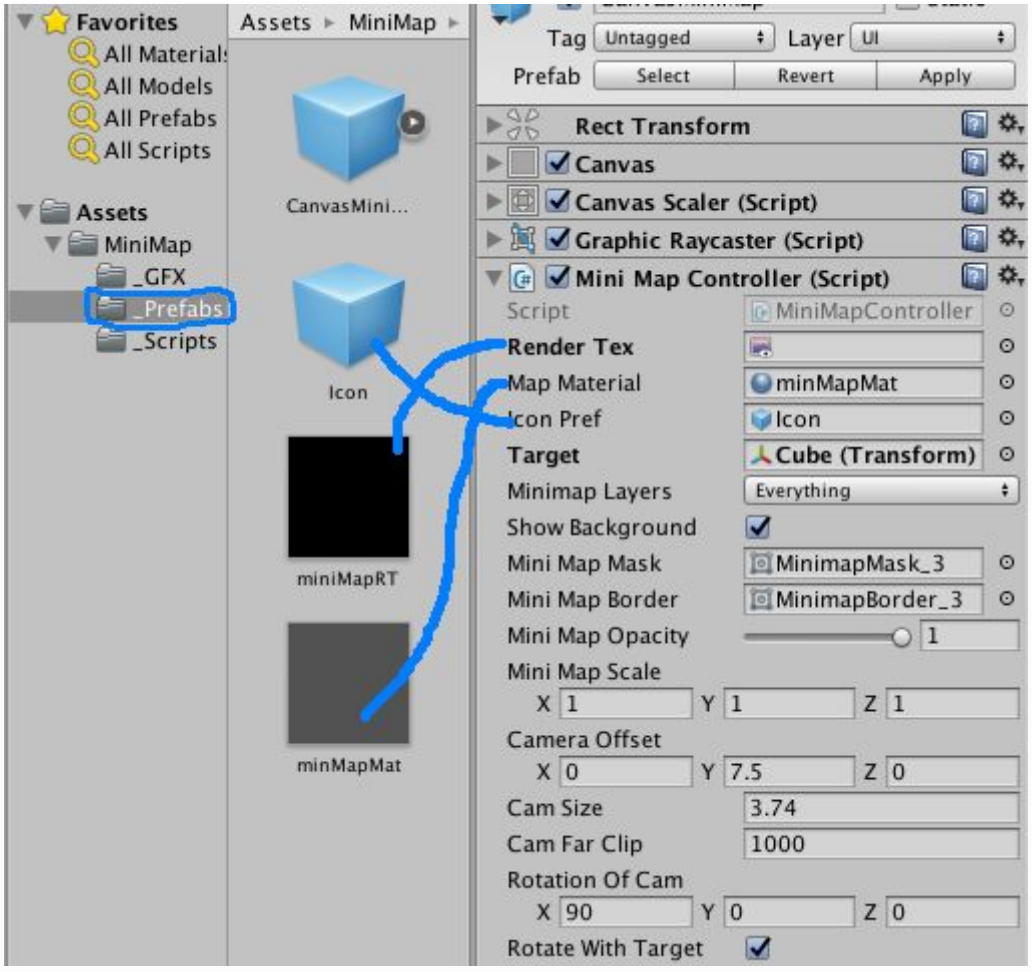
1. Import **Standard Assets/Characters/ThirdPersonCharacter** and **Standard Assets/Crossplatforminput.**
2. Run the demo scene in **Minimap/_Demo**

Installation:

1. Import the plugin package into your project.
2. Drag and drop the **CanvasMiniMap** prefab from ***Assets/MiniMap/_Prefab*** in the scene.
3. Set the **target** variable in the **MiniMapController** component in the **CanvasMiniMap** gameobject.
4. Add **MiniMapComponent** script from **Assets/MiniMap/_Scripts** to the gameobjects/prefabs, that you want to show on minimap.

Description:

1. MiniMapController:

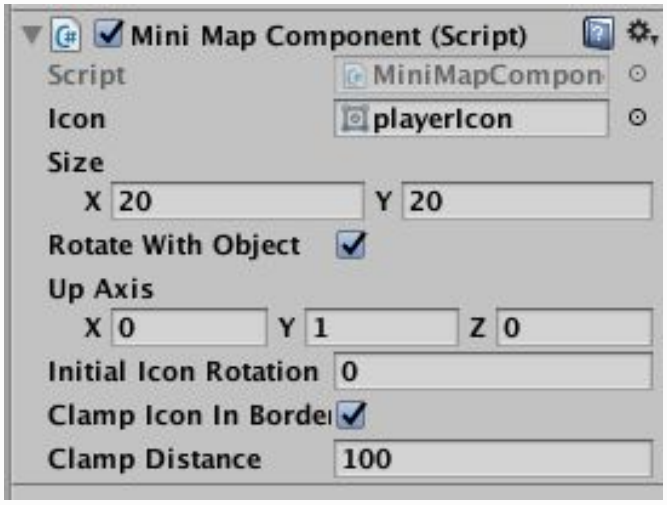


- **Render Tex:** Please assign "**MiniMap/_Prefabs/miniMapRT**" in this field.
- **Map Material:** Please assign "**MiniMap/_Prefabs/minMapMat**" in this field.

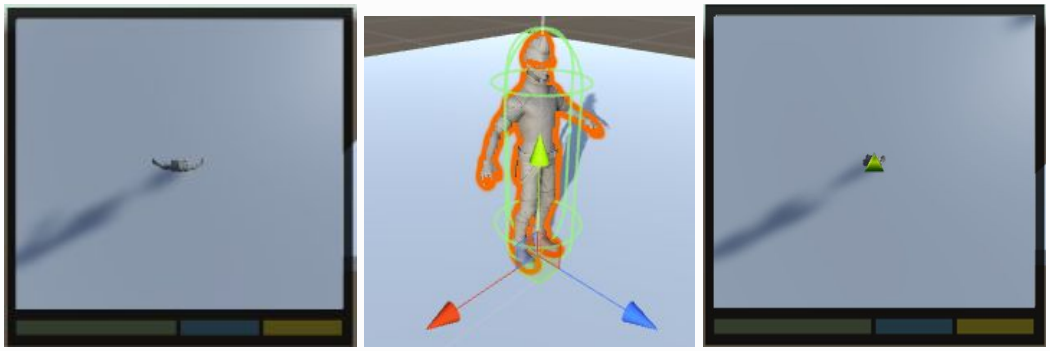
- **Icon Pref:** If this field is not assigned by default, then please assign "**MiniMap/_Prefabs/Icon**" prefab in this field.
- **Target:** The gameobject which will be in the centre of the minimap (the player).
- **Minimap Layers:** Layers that'll be shown in the minimap.
- **Mini Map Mask:** An image used to define the view area of the minimap. Samples are given..**(Image)**
- **Mini Map Border:** The border graphics of the Minimap. Samples are given..**(Image)**
- **Show Background:** If this is turned on, then image assigned in **Mini Map Border** variable will also be used as Minimap background.
- **Mini Map Opacity:** Controls the opacity of the Minimap.
- **Mini Map Scale:** Sets the size of the Minimap on screen.
- **Camera Offset:** Sets the **distance** of the minimap camera from the **Target**.
- **Camera FOV:** Controls the Field of view of the camera.
- **Cam Far Clip:** Controls the Far clip of the Minimap camera.**(Image)**
- **Rotation Of Cam:** The initial **rotation** of Minimap camera. Adjust this according to your scene.
- **Rotate With Target:** If this is **true**, the Minimap camera rotates with the **Target.** So, the player icon will point forward. Otherwise, the camera will not rotate with the **Target**.

2. MiniMapComponent:

This script needs to be added in the **gameobjects** or **prefabs** which needs to be shown on the Minimap.

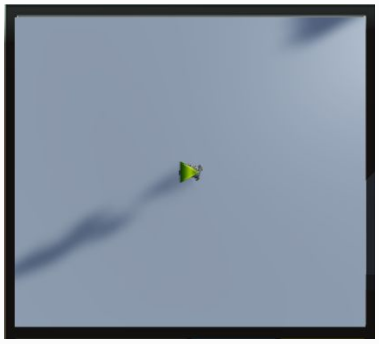


- **Icon:** Icon corresponding to the gameobject, which will be shown in the minimap to mark this gameObject.
- **Size:** Size of the icon.
- **Rotate With Object:** Set this true if you want the icon to be rotated with the direction of the gameObject.
- **Up Axis:** This option is included so that the user can define according to which axis of the gameObject, the icon needs to rotate. Values can be **-1,0 or 1** for each axis. Normally, a gameObject's **Y axis** is it's up axis and the icon rotates according to this axis. example:

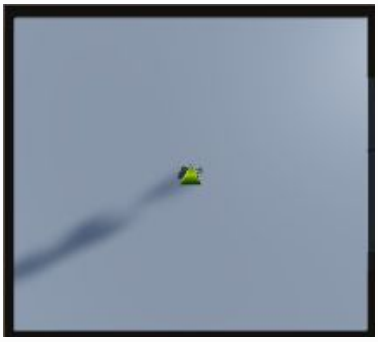


a. Minimap view b. Target axises(Up axis is Y) c. Minimap view after setting Up axis Y to 1.

- **Initial Icon Rotation:** Simple way to fix a mis-rotated icon. For example, if the graphical image you are using is like this,  then the minimap view will be like **(a)**. But it should be pointing upwards like **(b)**.



a. View with 0 rotation



b. View with 90 rotation

This can be achieved by changing the **Initial Icon Rotation** to 90.

- **Clamp Icon In Border:** Set this true, if you want to clamp the icon near the border of the minimap, when the object is out of minimap view range.

a. Clamp in border disable for enemy

b. Clamp in border enable for enemy

- **Clamp Distance:** If **Clamp Icon In Border** is checked, then this value comes into action.If the distance between the minimap **target** object and the current object is greater than **Clamp Distance,** then the icon disappears from the minimap. If this value is set to **0,** then the icon is always shown in the border when out of minimap view range.

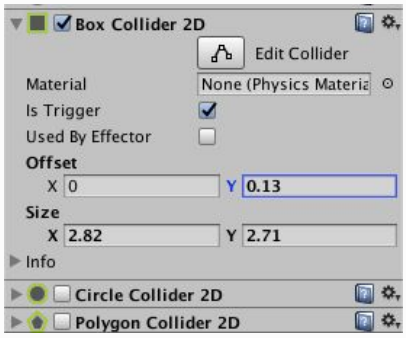
a. Enemy within clamp distance. b.Enemy far than clamp distance.

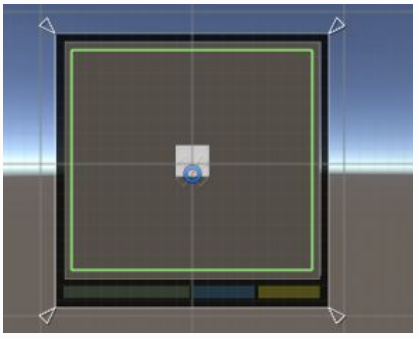
- **Setting up clamp border:** However, to clamp icons in the border we need to define the shape of the border using **Collider2D**. This has been given so users can use different shaped minimap. For this, first of all click **ShapeCollider** (image-**a**) in the scene view and you will see in the inspector (image-**b**)that by default 3 types

of collider are given**(Box,Circle and Polygon).** You'll need to enable only one and disable the others according to your minimap shape.

a. ShapeCollider gameobject in scene

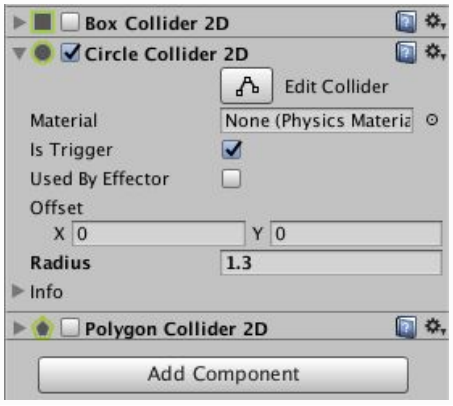
- Rectangular:

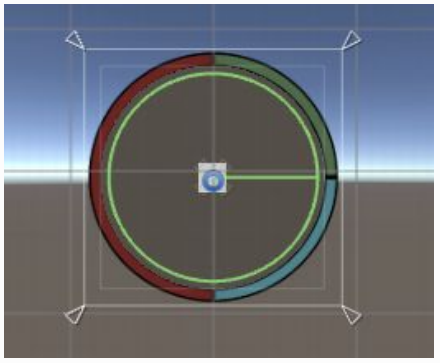


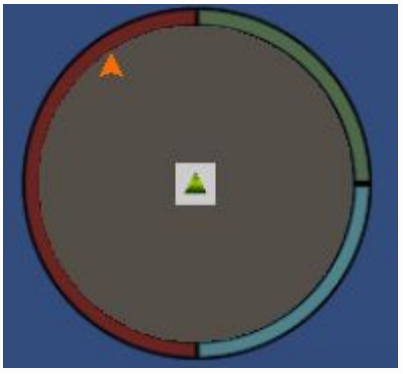


a. ShapeCollider inspector settings for rectangle b. Box Collider 2D marking the border in scene view c. Clamped icon in Game view

- Circular:

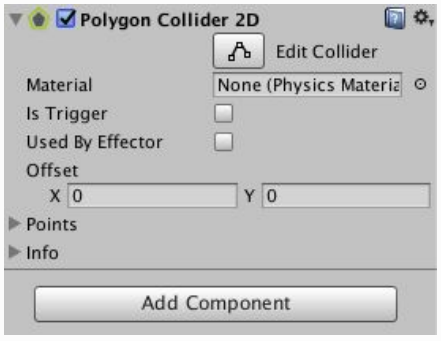


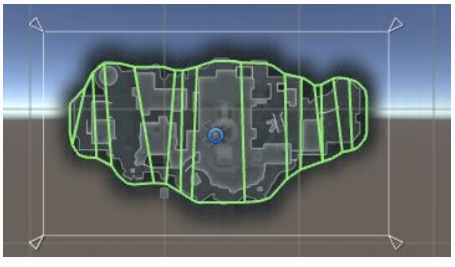




a. ShapeCollider inspector settings for Circle b. Circle Collider 2D marking the border in scene view c. Clamped icon in Game view

- Polygon(Random shaped minimap):





a. ShapeCollider inspector settings for Polygon b. Polygon Collider 2D marking the border in scene view c. Clamped icon in Game view

Support email:
upal.core2@gmail.com